Operating Systems Depth Exam - Spring 2018

UNIVERSITY OF WISCONSIN-MADISON
Computer Sciences Department

Instructions: There are **six questions** on this exam; you must **answer all** of them.

1. OS Protection

An OS provides security and protection either through hardware mechanisms, such as privileged mode bits and access bits in a page table, or through software permission checks in the kernel.

A. Explain how each of the following hardware features (i) contributes to providing protection and (ii) how the same function can be performed purely in software (assuming the other two features are still available):

- Address spaces
- Privileged mode of operation
- System calls

B. The OS kernel plays an integral role in enforcing security and protection as well. Explain the risk to security and protection if the OS kernel does not validate pointers passed to system calls.

C. The recent Spectre and Meltdown attacks allow user-mode programs to read, but not modify, privileged data, through architectural channels. What is the danger of giving user mode code this capability?

2. Distributed Storage Systems

Google's GFS and Amazon's Dynamo systems both provide storage for applications running in a data center. However, their designers made different choices regarding the tradeoffs between availability and consistency.

A. What is the availability and consistency guarantee offered by Dynamo, and why? For example, under what circumstances will Dynamo not be available or not offer consistent results?

B. What is the availability and consistency guarantee offered by GFS, and why? For example, under what circumstances will GFS not be available or not offer consistent results?

C. For either GFS or Dynamo: what would it mean for the system to provide strong consistency, and what would be the impact to the other properties of the system? Explain how you would change the design to achieve this.

3. Time, Clocks, Ordering

A. In a distributed system running on different physical machines, we often want to compare the execution order among events that occur on different machines. This comparison is usually not based on the physical clock maintained by each machine. Why?

B. Explain Lamport's logical clock scheme that a distributed system can use to determine the causal order among events. Here, the causal order should reflect message sending-receiving order (i.e., no message can be received before it is sent) and the execution order within each machine.

C. Lamport's scheme defines a partial ordering among events in a distributed program. For an arbitrary event E_i in process P_1 and event E_j in process P_2, Lamport's scheme cannot not always tell you whether Ei and Ej are ordered, i.e., whether either ($E_i \rightarrow E_j$) OR ($E_j \rightarrow E_i$) or there is no ordering relationship at all between E_i and E_j.
   i. Explain why Lamport's scheme cannot always establish an ordering between events.
   ii. How could you extend Lamport's scheme such that it could distinguish between these two cases above?

4. Monitors

While monitors were invented back in the early 1970's, they continue to be an important synchronization primitive, and are found (in some form) in languages like Java.

A. Monitors have two main synchronization primitives, the implicit monitor function/method entry lock and the condition variable. For each of these two mechanisms, describe their basic operations and semantics.

B. Proponents say that one of the advantages of monitor-based synchronization is that program's written with monitors will not have data races. However, the designers of the Mesa language created a notify/signal method that could be used outside of a monitor, calling this operation a naked notify.
   i. Why did the Mesa designers create the naked notify?
   ii. In what context was it typically used?

C. Describe how the use of naked notify can cause a synchronization problem in a Mesa program. What to you need to do to ensure that your program performs correctly?

5. Virtual Memory

Virtual memory represents one of the most basic abstractions provided by operating systems. And yet, aspects of the abstraction do not work as well today as they did in the past.

A. One potential problem with modern virtual memory systems is that they impose a high cost, due to TLB misses. Describe reasons why modern systems might suffer more from TLB misses than systems of the past, and discuss some potential solutions.

B. Another problem with VM systems is the cost of swapping pages to/from persistent storage is quite high, especially when considering older storage media like hard drives. Estimate the performance of a program as the percent of virtual-memory accesses that "hit" in memory changes from very high (all accesses are serviced from physical memory) to very low (all accesses must be serviced from the slower swap device). Be as quantitative as you can be.

C. To avoid the costs of swapping, in some datacenters, swapping pages out to disk is turned off entirely. Does virtual memory provide any benefits when swapping out is disabled? Discuss.

6. MapReduce

MapReduce is a popular paradigm for writing large-scale parallel data processing applications.

A. Describe the Map and Reduce phases of a MapReduce application. Make sure to discuss how data flows from inputs, through various stages of computation, and then to final outputs.

B. Describe how to implement a simple distributed grep application (which searches for a string within the input and outputs records that match).

C. Describe how to implement a parallel sorting application (which sorts each input record based on some key within the record).

D. What aspects of MapReduce are needed for grep and sort? Which aspects are not? If you were just interested in building those simple applications, how could you change/simplify MapReduce to make these applications run most efficiently?